COMPUTER SYSTEM AND METHOD FOR DECIDING EXISTENCE OF FPU IN COMPUTER SYSTEM

Publication number: JP2001147809
Publication date: 2001-05-29
Inventor: FARRALL GLENN; GEARTY MARGARET ROSE
Applicant: HITACHI LTD
Classification:
- international: ***G06F9/38;*** **G06F9/38**; (IPC1-7): G06F9/38
- european:
Application number: JP20000292743 20000926
Priority number(s): US19990411608 19991001

Report a data error here

Abstract of **JP2001147809**

PROBLEM TO BE SOLVED: To improve performance such as processing capability by closely synchronizing both arithmetic operations. SOLUTION: A computer system is provided with a single chip microcomputer having a central processing unit(CPU), a memory device connected with the CPU, and an interface 56 deformed so that the CPU can be connected with a floating point instruction processor (FPU) 54 and an FPU present signal connected from the interface 56 to the CPU, and a CPU present signal having a first state indicating the existence of an FPU in the single chip microcomputer to the CPU and a second state indicating the non-existence of any FPU in the single chip microcomputer to the CPU. Then, the single chip microcomputer transmits plural floating point instructions across the interface to an FPU 54 in response to the first state of the FPU present signal, and traps the plural floating point instructions in response to the second state of the signal.

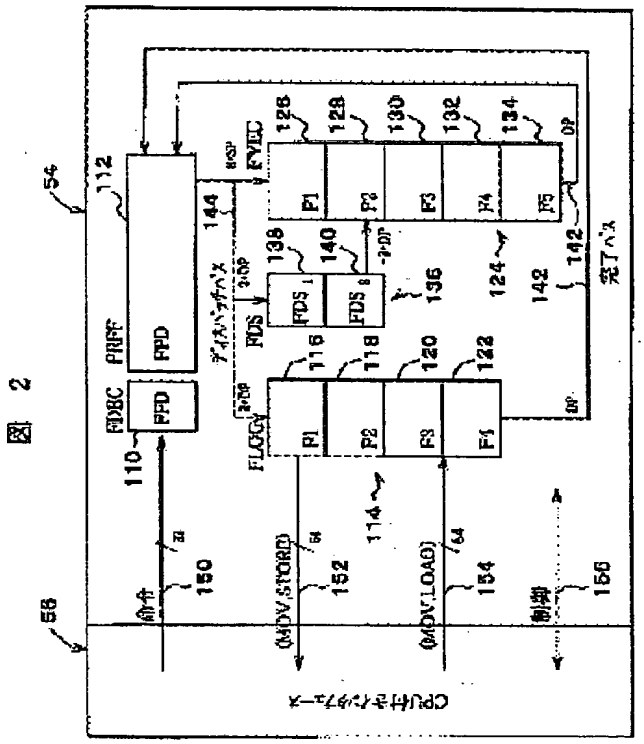


図 2

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2001－147809
（P2001－147809A）
(43)公開日　平成13年５月29日(2001.5.29)

(51)Int.Cl.⁷	識別記号	ＦＩ	テーマコード（参考）
Ｇ０６Ｆ　9/38	３７０	Ｇ０６Ｆ　9/38	３７０Ｃ
	３８０		３８０Ｂ

審査請求　未請求　請求項の数６　ＯＬ　（全 15 頁）

(21)出願番号　特願2000－292743(P2000－292743)

(22)出願日　平成12年９月26日(2000.9.26)

(31)優先権主張番号　０９／４１１６０８

(32)優先日　平成11年10月１日(1999.10.1)

(33)優先権主張国　米国（ＵＳ）

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(72)発明者　グレン・ファラル
イギリス国、ブリストル、ＢＳ41　９Ｊ
Ｑ、ロング・アシュトン・ロード　157

(72)発明者　マーガレット・ローズ・ギアティ
イギリス国、バス　ＢＡ１　７ＲＴ、バス
フォード、プランプ・レーン　３

(74)代理人　100080001
弁理士　筒井　大和

(54)【発明の名称】　コンピュータシステムとコンピュータシステム内でのＦＰＵの存在決定方法

(57)【要約】
【課題】　両演算を密接に同期化させて処理能力のような性能を向上させる。
【解決手段】　コンピュータシステムは、中央演算処理装置（ＣＰＵ）、ＣＰＵに接続したメモリー装置、及び、浮動小数点命令処理装置（ＦＰＵ）５４にＣＰＵを接続するために変形されたインタフェース５６を有する単一チップマイクロコンピュータと、インタフェース５６からＣＰＵに接続されたＦＰＵプレゼント信号と、単一チップマイクロコンピュータ内でのＦＰＵの存在をＣＰＵに示す第１状態と単一チップマイクロコンピュータ内でのＦＰＵの不存在をＣＰＵに示す第２状態とを有するＣＰＵプレゼント信号とを備え、単一チップマイクロコンピュータがＦＰＵプレゼント信号の第１状態に応答してインタフェースを横切って複数の浮動小数点命令をＦＰＵ５４に送り、信号の第２状態に応答して複数の浮動小数点命令をトラップする。

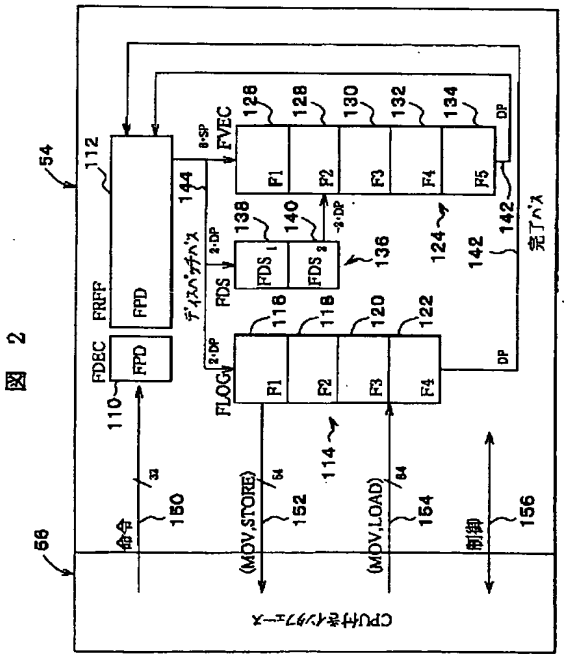


図２

【特許請求の範囲】

【請求項１】　コンピュータシステムであって、

中央演算処理装置（ＣＰＵ）、該ＣＰＵに接続したメモリー装置、及び、浮動小数点命令処理装置（ＦＰＵ）に前記ＣＰＵを接続するために変形されたインタフェースを有する単一チップマイクロコンピュータと、

前記インタフェースから前記ＣＰＵに接続されたＦＰＵプレゼント信号と、

前記単一チップマイクロコンピュータ内でのＦＰＵの存在を前記ＣＰＵに示す第１状態と前記単一チップマイクロコンピュータ内でのＦＰＵの不存在を前記ＣＰＵに示す第２状態とを有するＣＰＵプレゼント信号とを備え、

前記単一チップマイクロコンピュータが、前記ＦＰＵプレゼント信号の前記第１状態に応答して前記インタフェースを横切って複数の浮動小数点命令を前記ＦＰＵに送り、前記信号の前記第２状態に応答して複数の浮動小数点命令をトラップすることを特徴とするコンピュータシステム。

【請求項２】　請求項１記載のコンピュータシステムであって、

前記ＦＰＵプレゼント信号が前記第２状態であり、一つの浮動小数点命令がトラップされた時に、前記単一チップマイクロコンピュータが例外を提起することを特徴とするコンピュータシステム。

【請求項３】　コンピュータシステムであって、

中央演算処理装置（ＣＰＵ）、該ＣＰＵに接続されたメモリー装置、及び、浮動小数点命令処理装置（ＦＰＵ）に前記ＣＰＵを接続するために変形されたインタフェースを有する単一チップマイクロコンピュータと、

前記単一チップマイクロコンピュータ内でのＦＰＵの存在を前記ＣＰＵに示す指示手段と、

前記指示手段に敏感であり、前記指示手段に応答して前記単一チップマイクロコンピュータを制御する制御手段とを備えたことを特徴とするコンピュータシステム。

【請求項４】　請求項３記載のコンピュータシステムであって、

前記指示手段がＦＰＵプレゼント信号を有し、該ＦＰＵプレゼント信号が前記単一チップマイクロコンピュータ内でのＦＰＵの存在を示す第１状態と、前記単一チップマイクロコンピュータ内でのＦＰＵの不存在を示す第２状態とを備えたことを特徴とするコンピュータシステム。

【請求項５】　請求項４記載のコンピュータシステムであって、

前記指示手段が、前記第１状態内に前記ＦＰＵプレゼント信号のある時に、前記ＦＰＵに複数の浮動小数点命令を送り、前記第２状態内に前記ＦＰＵプレゼント信号のある時に、複数のＦＰＵプレゼント信号をトラップすることを特徴とするコンピュータシステム。

【請求項６】　中央演算処理装置（ＣＰＵ）と、該中央演算処理装置に接続されたメモリー装置と、前記ＣＰＵを浮動小数点命令処理装置（ＦＰＵ）に接続するように変形されたインタフェースとを備えた単一チップマイクロコンピュータを有するコンピュータシステム内で、ＦＰＵが前記コンピュータシステム内に存在するか否かを決定する方法であって、

前記方法が、

前記ＣＰＵを使って、前記単一チップマイクロコンピュータ内でのＦＰＵの存在をＣＰＵに示す第１状態と、前記単一チップマイクロコンピュータ内でのＦＰＵの不存在をＣＰＵに示す第２状態とを有するＦＰＵプレゼント信号を、前記インタフェースを横切って前記ＣＰＵに送り、

前記ＣＰＵを使って前記ＦＰＵプレゼント信号に応答し、前記単一チップマイクロコンピュータが前記ＦＰＵプレゼント信号の前記第１状態に応答して前記インタフェースを横切って前記ＦＰＵに複数の浮動小数点命令を送り、且つ前記ＦＰＵプレゼント信号の前記第２状態に応答して複数の浮動小数点命令をトラップするようにするステップを有することを特徴とするコンピュータシステム内でのＦＰＵの存在決定方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は一般にマイクロコンピュータ（マイクロコンピュータ／浮動小数点プロセッサのインタフェース及び方法）に関する。更に詳細には、本発明は中央処理実行ユニットと浮動小数点実行ユニットとを有する単一チップマイクロコンピュータに関する。

【０００２】

【従来の技術】システムオンチップデバイス（ＳＯＣ）、一般にマイクロコンピュータはよく知られている。これらのデバイスは一般的にプロセッサ（ＣＰＵ）と、１つ又はそれ以上のモジュールと、バスインタフェースと、メモリデバイスと、情報を交信するための１つ又はそれ以上のシステムバスとを含む。マイクロコンピュータに組み込まれることもあり得る１つのモジュールは、一般に浮動小数点ユニット、即ちＦＰＵと言われる浮動小数点コプロセッサである。浮動小数点ユニットは、非整数に関係する命令を実行するために用いられる。一般に、非整数は、２つの部分、即ち指数と有効数とに分割されたコンピュータワードとして表される。浮動小数点ユニットは特殊目的プロセッサであり、そのプロセッサは、数のこのような非整数表現に関係する算術演算を実行するように特に設計されている。

【０００３】完全に組込み又は埋込みされた浮動小数点ユニットを有するマイクロコンピュータは公知である。浮動小数点ユニットがマイクロコンピュータのＣＰＵ内に埋め込まれ、又はぴったりと統合された時、ＦＰＵとＣＰＵとは一般に幾つかの演算ブロックを共有する。従

って、ＦＰＵとＣＰＵとの間のインタフェースは、ハードウェア及びソフトウェアの両方において非常にぴったりと統合されている。このレベルの集積化が、高いスループット（処理能力）のような高い性能を提供するけれども、ＦＰＵの機能を望まない、又は必要としない顧客に対して販売するためにＦＰＵ無しのマイクロコンピュータバージョンを設計し、組立てることは困難なはずである。マイクロコンピュータ設計における多くの態様を変化させると、マイクロコンピュータからＦＰＵを除去することが非常に困難になる。そして、ある場合には、マイクロコンピュータからＦＰＵを除去することが重要な再設計努力を伴うことがある。

【０００４】個別のマイクロコンピュータと浮動小数点プロセッサとのシステムも同様に公知である。これらのシステムにおいては、マイクロコンピュータと浮動小数点ユニットとは一般に個別の集積回路チップであり、ＣＰＵとＦＰＵとの間で命令及びデータを交換するためにインタフェースが設けられている。ＣＰＵとＦＰＵとの間のインタフェースの一形式は緩衝配置を使用する。このタイプの配置では、ＣＰＵ及びＦＰＵ内での命令実行用のタイミング及び同期化必要条件を緩和でき、プロセッサ間での相対的なAloose（緩い）結合をもたらす。マイクロコンピュータにオプションとしてＦＰＵを提供することは容易である点で、このタイプのシステムは利点を有している。しかしながら、ＣＰＵとＦＰＵとの間の結合が緩いので、ＣＰＵ及びＦＰＵの演算が密接に同期化されていないために、処理能力のような性能は不利を招いている。

【０００５】

【発明が解決しようとする課題】しかしながら、ＣＰＵとＦＰＵとの間の結合が緩いので、ＣＰＵ及びＦＰＵの演算が密接に同期化されていないために、処理能力のような性能は不利を招いている。

【０００６】本発明の目的は、ＣＰＵ及びＦＰＵの演算を密接に同期化させて、処理能力のような性能を向上させることである。

【０００７】

【課題を解決するための手段】本発明の一態様によれば、単一チップマイクロコンピュータを有するコンピュータシステムが提供され、中央演算処理装置（ＣＰＵ）、該ＣＰＵに接続したメモリー装置、及び、浮動小数点命令処理装置（ＦＰＵ）に前記ＣＰＵを接続するために変形されたインタフェースを有する単一チップマイクロコンピュータと、前記インタフェースから前記ＣＰＵに接続されたＦＰＵプレゼント信号と、前記単一チップマイクロコンピュータ内でのＦＰＵの存在を前記ＣＰＵに示す第１状態と前記単一チップマイクロコンピュータ内でのＦＰＵの不存在を前記ＣＰＵに示す第２状態とを有するＣＰＵプレゼント信号とを備え、前記単一チップマイクロコンピュータが、前記ＦＰＵプレゼント信号の前記第１状態に応答して前記インタフェースを横切って複数の浮動小数点命令を前記ＦＰＵに送り、前記信号の前記第２状態に応答して複数の浮動小数点命令をトラップする。

【０００８】本発明の別の態様によれば、前記ＦＰＵプレゼント信号が前記第２状態であり、一つの浮動小数点命令がトラップされた時に、前記単一チップマイクロコンピュータが例外をレイズ（提起）する。

【０００９】本発明の別の態様によれば、中央演算処理装置（ＣＰＵ）、該ＣＰＵに接続されたメモリー装置、及び、浮動小数点命令処理装置（ＦＰＵ）に前記ＣＰＵを接続するために変形されたインタフェースを有する単一チップマイクロコンピュータと、前記単一チップマイクロコンピュータ内でのＦＰＵの存在を前記ＣＰＵに示す指示手段と、前記指示手段に敏感であり、前記指示手段に応答して前記単一チップマイクロコンピュータを制御する制御手段とを備えた。

【００１０】本発明の別の態様によれば、コンピュータシステムは単一チップマイクロコンピュータと、コンピュータシステム内でＦＰＵの存在を決定する方法とを有し、単一チップマイクロコンピュータが中央処理装置（ＣＰＵ）と、中央処理装置に結合されたメモリユニットと、ＣＰＵを浮動小数点命令処理ユニット（ＦＰＵ）に結合するように変更されたインタフェースとを含み、当該方法がＦＰＵを使ってインタフェースを横切ってＦＰＵプレゼント信号をＣＰＵに送り、ＣＰＵを使ってＦＰＵプレゼント信号に応答するステップを有し、当該ＦＰＵプレゼント信号が、単一チップマイクロコンピュータ内でＦＰＵの存在をＣＰＵに示す第一状態と、単一チップマイクロコンピュータ内でＦＰＵの不存在をＣＰＵに示す第２状態とを有し、単一チップマイクロコンピュータがＦＰＵプレゼント信号の第１状態に応答してインタフェースを横切って浮動小数点命令をＦＰＵに送り、ＦＰＵプレゼント信号の第２状態に応答して浮動少数点命令をトラップする。

【００１１】本発明の別の態様によれば、前記指示手段を含むコンピュータシステムは、ＦＰＵプレゼント信号を有し、該ＦＰＵプレゼント信号が前記単一チップマイクロコンピュータ内でのＦＰＵの存在を示す第１状態と、前記単一チップマイクロコンピュータ内でのＦＰＵの不存在を示す第２状態とを備えた。

【００１２】本発明の別の態様によれば、前記指示手段を含むコンピュータシステムは、前記第１状態内に前記ＦＰＵプレゼント信号のある時に、前記ＦＰＵに複数の浮動小数点命令を送り、前記第２状態内に前記ＦＰＵプレゼント信号のある時に、複数のＦＰＵプレゼント信号をトラップする。

【００１３】本発明の別の態様によれば、中央演算処理装置（ＣＰＵ）デコーダパイプステージを備えたＣＰＵ実行パイプラインと、浮動小数点ユニット（ＦＰＵ）デ

コーダパイプステージを備えたＦＰＵ実行パイプラインとを有するコンピュータシステム内で、前記方法が、ａ）第１命令を前記ＣＰＵデコーダパイプステージに伝送し、ｂ）前記第１命令を前記ＦＰＵデコーダパイプステージに伝送し、ｃ）前記第１命令が前記ＣＰＵデコーダパイプステージによって受け入れられたことを示す信号を生成し、ｄ）前記第１命令が前記ＦＰＵデコーダパイプステージによって受け入れられたことを示す信号を生成し、ｅ）ステップｄ）に応答して第２命令を前記ＣＰＵデコーダパイプステージに伝送し、ｆ）ステップｃ）に応答して第２命令を前記ＦＰＵデコーダパイプステージに伝送するステップを有する。

【００１４】本発明の別の態様によれば、コンピュータシステムは、ステップｄ）内に前記信号を発生させるまで、前記第１命令を前記ＣＰＵデコーダパイプステージに再び送るステップを更に有する。

【００１５】本発明の別の態様によれば、コンピュータシステムは、ステップｃ）内に前記信号を発生させるまで、前記第１命令を前記ＦＰＵデコーダパイプステージに再び送るステップを更に有する。

【００１６】本発明の別の態様によれば、コンピュータシステムは、中央処理装置（ＣＰＵ）実行パイプラインと、浮動小数点ユニット（ＦＰＵ）実行パイプラインとを有し、前記ＣＰＵパイプラインが複数のパイプステージを含み、そして前記ＦＰＵパイプラインが複数のパイプステージを含み、前記ＣＰＵパイプライン内の各ＣＰＵパイプステージが前記ＦＰＵパイプライン内で対応するパイプステージを有し、前記ＣＰＵパイプライン及びＦＰＵパイプラインの動作を同期化する方法であって、前記方法が、ａ）第１ＣＰＵパイプステージ内で一つの命令を受け取り、ｂ）対応する第１ＦＰＵパイプステージ内で前記命令を受け取り、ｃ）前記第１ＣＰＵパイプステージ内で前記命令を処理し、ｄ）前記第１ＦＰＵパイプステージ内で前記命令を処理し、ｅ）前記命令が前記第１ＣＰＵパイプステージによって処理されると共に、前記ＣＰＵパイプライン内で第２パイプステージの進行を準備することを示す第１信号を、前記第１ＣＰＵパイプステージによって生成し、ｆ）前記命令が前記第１ＦＰＵパイプステージによって処理されると共に、前記ＦＰＵパイプライン内で第２パイプステージの進行を準備することを示す第２信号を、前記第１ＦＰＵパイプステージによって生成し、ｇ）前記第１ＣＰＵパイプステージからの前記命令を前記ＣＰＵパイプライン内の前記第２パイプステージに送り、ｈ）前記第１ＦＰＵパイプステージからの前記命令を前記ＦＰＵパイプライン内の前記第２パイプステージに送るステップを有し、ｉ）前記ＣＰＵパイプライン内の前記第２パイプステージが、前記第２信号に応答して前記命令を前記ＣＰＵパイプライン内の第３パイプステージに送り、ｊ）前記ＦＰＵパイプライン内の前記第２パイプステージが、前記第１信号に応答して前記命令を前記ＦＰＵパイプライン内の第３パイプステージに送る。

【００１７】本発明の別の態様によれば、前記ＣＰＵパイプライン内の前記第２パイプステージが第２信号に更に応答して、前記ＣＰＵパイプライン内の前記第２パイプステージが、前記第１ＦＰＵパイプステージから別の第２信号を受け取るまで、前記ＣＰＵパイプライン内の前記第３パイプステージに複数命令を送るのを防止するような方法が提供される。

【００１８】本発明の別の態様によれば、前記ＦＰＵパイプラインが第１信号に更に応答して、前記ＦＰＵパイプライン内の前記第２パイプステージが、前記第１ＣＰＵパイプステージから別の第１信号を受け取るまで、前記ＦＰＵパイプライン内の前記第３パイプステージに複数命令を送るのを防止するような方法が提供される。

【００１９】本発明の別の態様によれば、コンピュータは、複数のパイプステージを含む中央処理装置（ＣＰＵ）実行パイプラインと、複数のパイプステージを含む浮動小数点ユニット（ＦＰＵ）実行パイプラインと、ＦＰＵパイプステージによって与えられた制御信号に応答して第１ＣＰＵパイプステージから第２ＣＰＵパイプステージへの複数命令の伝送を制御する第１手段と、ＣＰＵパイプステージによって与えられた制御信号に応答して第１ＦＰＵパイプステージから第２ＦＰＵパイプステージへの複数命令の伝送を制御する第２手段とを備え、前記ＣＰＵパイプライン内の各ＣＰＵパイプステージが前記ＦＰＵパイプライン内で対応するパイプステージを有する。

【００２０】本発明の別の態様によれば、制御する前記第１手段が、複数命令の伝送を可能にする第１状態と、複数命令の伝送を不可能にする第２状態とを有するトークン信号である。

【００２１】本発明の別の態様によれば、前記第１ＣＰＵパイプステージが、前記トークン信号の前記第１状態に応答して一つの命令を伝送する。

【００２２】本発明の別の態様によれば、前記第１ＣＰＵパイプステージが、一つの命令を伝送した時に、前記トークン信号をキャンセルする信号を生成する。

【００２３】本発明の別の態様によれば、前記第１ＦＰＵパイプステージが、前記トークン信号の前記第１状態に応答して一つの命令を伝送する。

【００２４】本発明の別の態様によれば、前記第１ＦＰＵパイプステージが、一つの命令を伝送した時に、前記トークン信号をキャンセルする信号を生成する。

【００２５】本発明の別の態様によれば、それぞれ複数のパイプステージを備えた中央演算処理装置（ＣＰＵ）実行パイプラインと、浮動小数点ユニット（ＦＰＵ）実行パイプラインとを有するコンピュータシステム内で、各ＣＰＵパイプステージが前記ＦＰＵ実行パイプライン内で対応するパイプステージを有し、前記ＣＰＵ実行パ

イプラインと前記ＦＰＵ実行パイプラインとの動作を同期化する方法であって、　前記方法が、ａ）前記ＣＰＵパイプライン内の各パイプステージに複数命令を提供し、ｂ）前記ＦＰＵパイプライン内でそれぞれの対応するパイプステージに前記命令を提供し、ｃ）前記ＣＰＵパイプライン内で前記命令を実行し、ｄ）前記ＦＰＵパイプライン内で前記命令を実行し、ｅ）停動状態に応答して前記ＣＰＵパイプラインを停動し、ｆ）前記ＣＰＵパイプラインを停動した後に、パイプステージの所定個数だけ前記ＦＰＵユニットパイプラインを停動し、ｇ）ステップｆ）に応答して前記浮動小数点処理装置パイプラインの実行状態を記憶し、ｈ）停動状態を解除して前記ＣＰＵ実行パイプラインを再開始し、ｉ）再開始時にステップｇ）内に記憶された前記データを前記ＣＰＵパイプラインに与え、ｊ）前記ＣＰＵパイプラインの再開始後に、所定個数のパイプステージで前記ＦＰＵパイプラインを再開始するステップを有する。

【００２６】本発明の別の態様によれば、ステップｇ）が、前記ＦＰＵパイプライン内の各パイプステージの実行結果を記憶することを更に含む方法が提供される。

【００２７】本発明の別の態様によれば、　前記所定個数のパイプステージが一つのパイプステージを有する方法が提供される。

【００２８】図面では、参照用としてここに組込まれ、同様な要素が同様な特性に与えられる図面が以下に与えられている。

【００２９】

【発明の実施の形態】図１は、本発明による単一チップマイクロコンピュータ５０を示す。マイクロコンピュータ５０は、コンピュータ内でのオペレーション（演算）を実行するための中央処理装置コア５１を含む。整数中央処理装置（ＣＰＵ）５２と任意浮動小数点処理装置（ＦＰＵ）５４とがＣＰＵコア５１の一部として装備される。詳細に後述されるインタフェース５６は、整数ＣＰＵ５２とＦＰＵ５４との間でデータ、命令、及び制御信号を交換する機構を備える。また、ＣＰＵコア５１は、例えば命令フェッチユニットおよびロードストアユニットのような他のモジュールも含む。本記述において、ＣＰＵ５２は、整数演算を実行するＣＰＵコア５１の一部分に関係している。ＣＰＵコア５１は、データリンク６０を介してシステムバス５８に結合されている。システムバス５８は、システムバスに添付されたモジュール及びインタフェースの間でデータ、命令、及び制御信号を交換するための通路（パスウェイ）を備える。

【００３０】オフチップランダムアクセスメモリにインタフェースを提供するＲＡＭインタフェース６２は、データリンク６４を介してシステムバス５８に結合される。オフチップ読取り専用メモリにアクセスを提供するＲＯＭインタフェース６６は、データリンク６８を介してシステムバス５８に結合される。他のシステムバスモジュール７０は、データリンク７２によってシステムバス５８に結合される。

【００３１】デバッグインタフェースを含むデバッグモジュール７４は、データリンク７６を介してシステムバス５８に結合される。デバッグモジュール７４は、データリンク８０を介してＣＰＵコア５１からデバッギングデータを受け取る。デバッグモジュール７４はデバッグリンク８２を介してオフチップインタフェースを供給する。そのデバッグリンク８２によってマイクロコンピュータ５０が外部装置又はソフトウェアをインタフェースで連結可能となる。

【００３２】また、マイクロコンピュータ５０は、データリンク８６を介してシステムバス５８に結合されたシステムバスアービタ８４を含む。システムバスアービタ８４は、システムバス５８を介してデータトラヒックの流れを制御する。システムバス８４は、データリンク８８を介して、例えばシステムバスウォッチポイントをトリガ（誘発）するようなデバッギング情報をデバッグモジュール７４へ送る。

【００３３】また、マイクロコンピュータ５０は周辺コンポネントバス９０も含む。周辺コンポネントバスアービタ９２は、周辺コンポネントバス９０を介してデータフローを制御し、データリンク９４を介して周辺コンポネントバス９０に結合され、データリンク９６を介してシステムバス５８にインタフェースを提供する。

【００３４】周辺コンポネントバスモジュール９８は、データリンク１００を介して周辺コンポネントバス９０に結合可能である。データリンク１０４を介して周辺コンポネントバス９０に結合された周辺コンポネントバスインタフェース１０２は、オフチップコンポネント用のインタフェースを周辺コンポネントバス９０に提供する。

【００３５】図２は、図１に示したＦＰＵ５４と、インタフェース５６との更に詳細なブロック図である。ＦＰＵ５４は多くの機能モジュールを含む。モジュール１１０は浮動小数点ユニットデコーダ及びパイプ制御ブロックであり、それらはインタフェース５６を介して送られたＣＰＵ５２からの３２ビット命令を復号する。モジュール１１２は浮動小数点ユニットレジスタファイル及び転送ネットワークである。モジュール１１４は、それぞれ１１６，１１８，１２０，１２２と番号付けられた実行パイプラインステージＦ１，Ｆ２，Ｆ３，Ｆ４から成り、共同実行済みＣＰＵ命令を実行すると共に、レジスタアクセスを制御するための浮動小数点論理実行モジュールである。モジュール１２４は、それぞれ１２６，１２８，１３０，１３２，１３４と番号付けられた実行パイプラインＦ１，Ｆ２，Ｆ３，Ｆ４，Ｆ５から成り、次の演算を実行するための浮動小数点ベクトル及び基礎計算ユニットである。即ち、その演算は、計算、ブロッキング計算、ベクトル計算、ベクトル計算、ブロッキング

ベクトル計算、タイプ変換、及び、多項式計算の演算である。モジュール１３６は、それぞれ１３８，１４０と番号付けられた実行パイプステージＦＤＳ１及びＦＤＳ２から成り、例えば除算及び平方根の演算のような非ブロッキング計算の演算を実行するための浮動小数点除算及び平方根実行ユニットである。完了バス１４２とディスパッチバス１４４とはモジュール１１４，１２４，１３６をモジュール１１２に結合する。

【００３６】当該技術分野における当業者が理解できることは、図面を簡素化するために、以下の説明において、図示された論理の演算に必要なクロック信号が図示されていないことである。しかしながら、当該技術分野の当業者は、いつどこで適切なクロック信号を適用して所望の機能を達成したかを理解できるだろう。

【００３７】本発明の特徴によれば、ＦＰＵ５４がＣＰＵコア５１のすべてを備えた着脱自在の部分であるように設計されている。従って、インタフェース５６を介したＣＰＵ５２とＦＰＵ５４の間のデータ移動は、データ移動用の３２ビット命令１５０と、二つの６４ビットバス１５２，１５４とに制限される。また、制御信号インタフェース１５６は、ＣＰＵ５２とＦＰＵ５４の間における命令の実行を制御し、同期化するために装備される。

【００３８】図３は、実行パイプラインの構造と、ＣＰＵ５２及びＦＰＵ５４内での実行パイプラインにおける様々なパイプステージの間の関係とを示す。ＣＰＵ５２は実行パイプライン１６０を含む。ＦＰＵ５４は実行パイプライン１６２を含む。各パイプライン１６０，１６２は多数のパイプステージを含む。ＣＰＵ５２とＦＰＵ５４とは命令フェッチパイプステージ１６４とプリデコーダパイプステージ１６６と分けている。ＣＰＵのパイプライン１６０は、デコーダパイプステージ１６８、三つの実行パイプステージ１７０，１７２，１７４、及び書き戻しパイプステージ１７６を含む。ＦＰＵのパイプライン１６２は、浮動小数点デコーダパイプステージ１７８と、五つの実行パイプステージ１２６，１２８，１３０，１３２，１３４と、浮動小数点書き戻しステージ１８０とを含む。その浮動小数点書き戻しステージ１８０は、ＣＰＵ５２へ送り返すために、浮動小数点ユニットにおける実行パイプライン１６２の結果をモジュール１１２に送る。

【００３９】演算の間に、命令が、実行用のＣＰＵパイプライン１６０及びＦＰＵパイプライン１６２の両方に同時に送られる。ＣＰＵパイプライン１６０及びＦＰＵパイプライン１６２によって実行される二つのタイプの命令が存在する。命令の第１カテゴリは、ＣＰＵパイプライン１６０内で完全に実行し、ＦＰＵパイプライン１６２からの完了に関して一切の寄与を必要としない純粋ＣＰＵ命令である。更に詳細に後述するように、ＣＰＵパイプライン１６０とＦＰＵパイプライン１６２とは密接に結合され、従って、純粋ＣＰＵ命令がＣＰＵパイプライン１６０内で実行されると、命令イメージはＦＰＵ１６２パイプライン内で実行される。ＣＰＵパイプライン１６０内で実行した純粋ＣＰＵ命令の場合には、ＦＰＵパイプライン１６２内での当該命令のイメージがバブルである。

【００４０】ＣＰＵパイプライン１６０及びＦＰＵパイプライン１６２内で実行する命令の第２カテゴリがＦＰＵ命令である。全てのＦＰＵ命令はこのグループ内である。例外項目と完了状態とを収集するだけであるならば、全てのＦＰＵ命令は、ある程度までＣＰＵパイプライン１６０内で命令イメージとして実行しなければならない。第１サブグループのＦＰＵ命令は、データ交換を備えた結合ＣＰＵ－ＦＰＵ命令である。これらの命令は、ＣＰＵパイプライン１６０とＦＰＵパイプライン１６２との間でのデータ交換を、ＦＰＵからＣＰＵへ、又はＣＰＵからＦＰＵへ伴う。第２サブグループのＦＰＵ命令は、データ交換なしの接合ＣＰＵ－ＦＰＵ命令である。これらの命令がＦＰＵパイプライン内で完全に実行し、ＣＰＵパイプライン１６０がその命令とだけ関係して例外情報及び完了状態を収集する。ＦＰＵパイプライン１６２とＣＰＵパイプライン１６０との間でデータ交換なしの結合ＣＰＵ－ＦＰＵ命令がＦＰＵパイプライン１６２内で実行すると、浮動小数点位置ホルダがＣＰＵパイプライン１６０を介し命令イメージとして実行する。その命令イメージは例外項目を収集し、パイプラインの同期化を維持する。データ交換を備えた接合ＣＰＵ－ＦＰＵ命令がＦＰＵパイプライン１６２内で実行すると、ＦＰＵ命令もＣＰＵパイプライン１６０内で命令イメージとして実行するので、パイプラインは同期状態のままである。

【００４１】本発明の特徴は、ＦＰＵパイプライン１６２とＣＰＵパイプライン１６０との間での実行における密接な結合と同期化とを維持することである。２つのパイプラインの間での密接な結合と同期化とを維持することは幾つかの利点を有する。ＦＰＵパイプライン１６２とＣＰＵパイプライン１６０との間での密接な同期化の維持によって、マイクロコンピュータ５０が精密な例外モデルの維持を可能とすることは、重要な利点である。精密な例外モデルが意味していることは、マイクロコンピュータ５０内の幾つかのハードウェア又はソフトウェアの問題に起因して例外が生成された時に、マイクロコンピュータ５０の実行の状態がエラー発生の時点で明らかになるように、命令が実行及び終了しなければならないことである。これによって例外発生の時点で種々の構成要素の状態が検査され、正しい対策の採用されるのを可能にする。精密な例外モデルが維持されないならば、エラー発生時に次のような状態を決定することが困難になる。即ち、マイクロコンピュータの様々の構成要素がエラー発生時に存在し、問題の追跡及び訂正を非常に困

難にすることがあり得ることである。

【００４２】本発明の別の特徴は、ＦＰＵ５４がオプションである。更に詳細に後述されるように、ＦＰＵ５４とＣＰＵ５２との間のインタフェース５６は次のように設計されている。即ち、特定バージョンのマイクロコンピュータ５０からＦＰＵを削除することによって、マイクロコンピュータの重要な再設計が必要とされないことである。回路の再設計又はソフトウェアの修正なしに、マイクロコンピュータ５０を含む単一集積回路から、ＦＰＵ５４を完全に削除することは簡単に可能である。

【００４３】従って、インタフェース５６によってＦＰＵ５４が、マイクロコンピュータ５０内でオプションであることを可能にするだけでなく、高水準の処理能力性能を提供可能とする。それから、同時にマイクロコンピュータ５０が演算の精密な例外モデルを維持することを可能にする間に、個別のマイクロコンピュータ及びコプロセッサが高水準の処理能力性能を提供する。

【００４４】図４は、ＣＰＵ５２とＦＰＵ５４との間で＊のインタフェース５６を示した更に詳細なブロック図である。下記の表１は、ＣＰＵ５２とＦＰＵ５４との間で通信に用いられる信号の集合を示す。AName' 欄は各制御信号の名前を与える。ADir' 欄は、信号がＦＰＵへの入力であるか又はＦＰＵからの出力であるかに関して各信号の方向を示す。ASrc' 欄は、ＣＰＧ、（クロック発生回路）、ＦＰＵ、命令フェッチユニット（ＩＦＵ）、及び、ロード／ストアユニット（ＬＳＵ）の間において、どのユニットが信号の源であるかを示す。ASize' 欄は信号内のビット数を示す。AStage Sent' 欄は、ＣＰＵ５４又はＣＰＵ５２におけるどのステージが信号を送るかを示す。ALatched by' 欄は、信号がインタフェース５６のＣＰＵ側でラッチされた（取り込まれた）か、又はインタフェース５６のＦＰＵ側でラッチされたかを示す。ADescription' 欄は各信号の説明を与える。

【００４５】

【表１】

【表１】

Name	Dir	Src	Size	Stage Sent	Latched by	Description
cpg_fpu_clk_en	in	CPG	1			FPU 用ｸﾛｯｸ停止
fpu_present	out	FPU	1	CPU		FPU が存在するかどうかを表示
ifu_sr_fd	in	IFU	1	W	CPU	SR 浮動小数点不能化ビット
ifu_fpu_inst_pd	in	IFU	28	PD	FPU	演算ｺｰﾄﾞ (ﾌﾟﾘﾃﾞｺｰﾄﾞｽﾃｰｼﾞで送られる)
ifu_fpu_inst_valid_pd	in	IFU	1	PD	FPU	演算ｺｰﾄﾞは有効 (ﾌﾟﾘﾃﾞｺｰﾄﾞｽﾃｰｼﾞにおいて) FPD で使用可能
ifu_fpu_pred_inst_pd	in	IFU	1	PD	FPU	伝送中の命令はﾌﾞﾗﾝﾁ予測経路上に在る
ifu_fp_go_dec	in	IFU	1	D	FPU	IFU デコーダｽﾃｰｼﾞにおける有効FP命令が進行可能（停動しない）
ifu_fpu_mispred_e2	in	IFU	1	E2	CPU	間違って予測された第２ﾌﾞﾗﾝﾁは CPUﾊﾞｲﾊﾟｽ内で解決
ifu_fpu_cancel_wb	in	IFU	1	W	CPU	WB における FPU/CPU 命令は連携 CPU 例外を有し、ﾊﾟｲﾌﾟﾗｲﾝはｷｬﾝｾﾙされなければならない (F4 から FPD へ戻る)
lsu_stall_e3	in	LSU	1	E3	FPU	E3 ｽﾃｰｼﾞﾊﾞｯｸは CPU において停動 (F4 においてのみ使用可能)
ifu_fpu_data_wb[63:0]	in	IFU	64	W	CPU	FLD, FMOV (F4 において使用可能) 用整数 CPU からのデータ
fpu_fp_go_dec	out	FPU	1	FPD	CPU	FPU デコーダｽﾃｰｼﾞにおける有効FP命令は進行可能
Fpu_dec_stall	out	FPU	1	FPD	CPU	FPU デコーダﾊﾞｯﾌｧは有効 FP 命令を有し FPD は内部で停動中、従って CPU からの新規命令は受入れ不能
Fpu_ifu_excep_f2	out	FPU	1	F2	CPU	FPU 例外が発生した
Fpu_lsu_data_f1[63:0]	out	FPU	64	F1	CPU	整数 CPU (E2 において使用可能) へのデータ
fpu_lsu_fcmp_f2	out	FPU	1	F2	CPU	FCMP 結果 (E3 において使用される)

【００４６】注意されるように、ＦＰＵ５４とＣＰＵ５２との間で通過する信号はラッチされる。ALatched by' 欄は、インタフェースのどちら側にラッチング回路が配置されているのかを示す。ＣＰＵ５２とＦＰＵ５４と

の間でのフライトの時間のために、ラッチング回路が必要である。

【００４７】ｆｐｕ＿ｐｒｅｓｅｎｔ信号は、ＦＰＵが存在するか否かをＣＰＵに示す。ＦＰＵが存在するならば、この信号がアサート（表明）され、ＦＰＵが利用可能であることをＣＰＵが認識する。この情況の下で、ＣＰＵはＦＰＵに命令を送る。ｆｐｕ＿ｐｒｅｓｅｎｔ信号がデザート（表明）されなかったならば、ＦＰＵが存在しないことをＣＰＵが認識する。この情況の下で、ＦＰＵ命令に遭遇したならば、ＣＰＵは当該命令をトラップし、例外を出現させる。従って、ＦＰＵの存在又は不在に応じて変化する唯一の信号がｆｐｕ＿ｐｒｅｓｅｎｔ信号である。

【００４８】ＦＰＵ５４の割り込みを抑制するために、浮動小数点不能化信号ｉｆｕ＿ｓｒ＿ｆｄが提供される。このフラグがＣＰＵの状態レジスタ（ＳＲ）に設定されると、ＦＰＵ５４が割り込みを抑制され、全ての浮動小数点命令がトラップされる。

【００４９】ここで、ＣＰＵパイプライン１６０及びＦＰＵパイプライン１６２を同期化する回路と信号とを示す図４を参照する。通常、ＣＰＵパイプライン１６０及びＦＰＵパイプライン１６２はロックステップ内で命令を実行する。即ち、例えばそれぞれ一対のＣＰＵパイプステージ１７０（又は１７２）、及び、ＦＰＵパイプステージ１２６（又は１２８）を経て進行する命令を実行する。更に詳細に後述されるように、パイプライン内には同期から消え去ることができ、実行継続前に再同期化の必要な三点が存在する。しかしながら、パイプライン間の最大スリップは、図に示す実施の形態における１つの命令又は、１つのパイプステージに限定される。けれども、ＦＰＵパイプライン１６２及びＣＰＵパイプライン１６０が、パイプラインの停動前に許容されたスリップ量において制限され、そしてパイプラインが、停動状態の解除時に相互に同期される。そのため、精密な例外モデルを実行できる。パイプラインにおいて同期化の喪失可能な点は、プリデコーダステージ１６６、デコーダ／Ｅ１－Ｆ１パイプステージ、及び、Ｅ３／Ｆ４パイプステージ内で発生する。これらの同期化メカニズムの各々が以下に後述される。

【００５０】ＣＰＵパイプライン１６０内の各パイプステージ１６８，１７０，１７２，１７４，１７６は、前のパイプステージからの計算結果を記憶するためにそれぞれのバッファ２２４，１７０Ａ，１７２Ａ，１７４Ａ，１７６Ａを有する。ＦＰＵパイプライン１６２内の各パイプステージ１７８，１２６，１２８，１３０，１３２，１３４，１８０は、前のパイプステージからの計算結果を記憶するためにそれぞれのバッファ２２６，１２６Ａ，１２８Ａ，１３０Ａ，１３２Ａ，１３４Ａ，１８０Ａを有する。

【００５１】信号がＣＰＵパイプライン１６０とＦＰＵパイプライン１６２との間のインタフェース５６を横切って移動するに要する時間（フライト時間）のために、そして幾らかの信号が１クロックサイクル内で遅れて到達するために、ＦＰＵから到達する信号用としてＣＰＵ側にラッチが設けられ、そしてＣＰＵから到達する信号用としてＦＰＵ側にラッチが設けられる。ＣＰＵ側はラッチ１７０Ｂ，１７２Ｂ，１７４Ｂ，１７４Ｃを含む。ＦＰＵ側はラッチ１２６Ｂ，２８４を含む。

【００５２】図４から図７に示す実施の形態によってＣＰＵ及びＦＰＵパイプラインが最大１つのパイプステージまでパイプライン相互の同期化から外れることを可能とする。しかしながら、本発明は、一回のパイプステージに限定されるのではなく、任意の所定個数のパイプステージを（又はゼロ回のパイプステージさえも）可能となる。即ち、各パイプラインを実行するデータ及び状態が記憶されているから、パイプラインの停動前に、所定個数のクロックサイクルによってパイプラインが同期から外れるのを可能とされる。パイプラインの再開始時に、一つのパイプライン内の幾つかのパイプステージからデータが、適切なタイミングで他のパイプラインに利用できる。そのため、データの損失なく停動以前と同じ関係にパイプラインを再同期化できる。その上、ＣＰＵ及びＦＰＵパイプラインが所定個数のクロックサイクルによって同期から外れるのを可能とすることにより、インタフェース５６を横切ってＣＰＵパイプラインとＦＰＵパイプラインとの間のフライト時間を埋め合わせている。

【００５３】次に、ＣＰＵプリデコーダステージ命令のバッファリング機構の動作を示す図５を参照する。回路におけるこのセクションはプリデコーダ論理回路２００を含み、そのプリデコーダ論理回路２００はラッチ２０２を介してＣＰＵ命令フェッチユニットから命令フェッチユニットデコーダの停動信号を受け取る。また、プリデコーダ論理２００は、ラッチ２０４を介して浮動小数点ユニットデコーダ１７８から浮動小数点ユニットデコーダの停動信号をも受け取る。浮動小数点ユニットデコーダ１７８が共有プリデコーダステージによって送られた次の信号を受信せず、且つラッチしない時はいつでも、ｆｐｕ＿ｄｅｃ＿ｓｔａｌｌ信号が生成される。ＣＰＵ５２の命令フェッチユニットが何等かの理由で停動されるたびに、ｉｆｕ＿ｄｅｃ＿ｓｔａｌｌ信号が生成される。

【００５４】マルチプレクサ２０６は、プリデコーダバッファ２０８に結合された多くの入力を有する。接続部２１０によってマルチプレクサ２０６の出力がプリデコーダバッファ２０８、プリデコーダ２１２、又はマルチプレクサ２１４に送られるのを可能とする。プリデコーダ２１２の出力は接続部２１６を介してマルチプレクサ２１８に送られる。マルチプレクサ２１４，２１８はそれぞれ出力２２０，２２２を有し、その出力２２０，２

２２は命令フェッチユニットデコーダバッファ２２４とＦＰＵデコーダバッファ２２６にそれぞれ結合される。バッファ２２４，２２６は、デコーダ１６８，１７８によってデコーダ中の命令を保持するために役立つ。バッファ２２４は出力２２７を有し、その出力２２７によってバッファ２２４内の命令がマルチプレクサ２１８へ戻って再循環可能となる。同様な方法で、バッファ２２６は出力２２８を有し、その出力２２８によってバッファ２２６内の現行命令がマルチプレクサ２１４に戻って再循環可能となる。ｉｆｕ＿ｄｅｃ＿ｓｔａｌｌ信号が何等かの理由によってアサート（表明）されるならば、マルチプレクサ２１８は停動状態の解除まで命令の選択と再循環とを続けるだろう。同様の方法で、ｆｐｕ＿ｄｅｃ＿ｓｔａｌｌ信号がアサート（表明）されるならば、マルチプレクサ２１４は停動状態の解除まで命令２２８をバッファ２２６内に再循環し続けるだろう。

【００５５】既に述べたように、ＣＰＵ命令フェッチユニットからの命令は、実行のためにＣＰＵパイプライン１６０とＦＰＵパイプライン１６２との両方に送られる。パイプラインが新しい命令の受け入れ準備をすると直ぐに、論理がプリデコーダステージ命令をパイプラインに送る。しかし、現行命令が他のパイプライン（ＣＰＵ又はＦＰＵ）によって受け入れられるまで、論理が別の命令を送らない。図５に示すプリデコーダステージ論理が請け合っていることは、ＣＰＵパイプライン１６０のデコーダステージ１６８とＦＰＵパイプライン１６２のデコーダステージ１７０とが任意のクロックサイクルの間に同期から外れた多くとも一つの命令であり得ることである。現行命令が両方のパイプラインによって受け入れられ又は扱われることがないことを保証するために、プリデコーダ論理２００は次の機能を実行する。即ち、

```
select_PDbuf = ～(IFU_taken & FPU_taken)
IFU_taken = ～ifu_dec_stall_q | IFU_taken_earlier_
q
FPU_taken = ～fpu_dec_stall_q | FPU_taken_earlier_
q
IFU_taken_earlier_d = IFU_taken & ～new_PD_inst_va
lid
FPU_taken_earlier_d = FPU_taken & ～new_PD_inst_va
lid
new_PD_inst_valid = IFU_taken & FPU_taken & a_new_
PD_inst_is_avail-able
```

である。ここに、ｉｆｕ＿ｄｅｃ＿ｓｔａｌｌ＿ｑはラッチ２０２によって出力された信号であり、ｆｐｕ＿ｄｅｃ＿ｓｔａｌｌ＿ｑはラッチ２０４によって出力された信号であり、ＩＦＵ＿／ＦＰＵ＿ｔａｋｅｎ＿ｅａｒｌｉｅｒ＿ｑはＩＦＵ＿／ＦＰＵ＿ｔａｋｅｎ＿ｅａｒｌｉｅｒ＿ｄ信号のラッチされたバージョンである。

【００５６】両方のパイプラインが実際にＡstall信号（ｉｆｕ＿ｄｅｃ＿ｓｔａｌｌ及びｆｐｕ＿ｄｅｃ＿ｓｔａｌｌ）を生成するだけであるので、これらの信号はＡtaken信号に変換される。この変換の達成は、ラッチ２０２，２０４内の停動信号をラッチし、プリコード論理２００への信号の供給前にラッチ出力を反転してｉｆｕ＿ｄｅｃ＿ｓｔａｌｌ＿ｑ信号とｆｐｕ＿ｄｅｃ＿ｓｔａｌｌ＿ｑ信号とを供給することによって行われる。

【００５７】プリデコーダバッファ２０８とマルチプレクサ２０６との間の接続から分かるように、プリデコーダステージ命令は追加クロックサイクル用のプリデコーダバッファ２０８内に常に記憶される。これが請け合っていることは、ＣＰＵパイプラインデコーダ１６８とＦＰＵパイプラインデコーダ１７８とが同一命令を受け入れるまで、プリデコーダバッファ２０８のコンテンツがプリデコーダステージ内で常に利用可能なことである。図５に示す論理の結果として、ＦＰＵ又はＩＦＵからの停動状態にもかかわらず、デコーダステージ１６８，１７８が同期化のたった一つの命令であり、同一命令が同時刻にＣＰＵデコーダステージ１６８とＦＰＵデコーダステージ１７８とを終了させ、それにより両方のパイプラインがこの点で同期化されるだろう。

【００５８】さて、ＣＰＵデコーダ／ＦＰＵデコダーＥ１／Ｆ１同期化論理の論理的ブロック図を示す図６を参照する。

【００５９】ひとたび命令がＣＰＵパイプライン１６０とＦＰＵパイプライン１６２とに与えられると、２つのパイプライン内の異なったデコーダステージ停動状態に起因して、同期化が即座に失われる。同期化におけるこの損失を克服するためには、Ａgo－tokenパッシング機構が用いられて両パイプラインを再同期化した後に、同一浮動小数点命令の２つのイメージがそれぞれのパイプステージ１７０，１２６を離れる。各パイプラインが、有効浮動小数点命令を復号した時に、ｇｏ－ｔｏｋｅｎを他のパイプラインに送り、任意のデコーダステージ停動状態のために停動されない。それから、このｇｏ－ｔｏｋｅｎは他のパイプライン内でラッチされ、他のパイプライン内で同一命令のイメージ用のゲート状態として用いられてパイプステージ１７０，１２６を越えて進行する。浮動小数点命令のイメージがパイプステージ１７０又は１２６を離れる時に、新しいｇｏ－ｔｏｋｅｎを受け取るまでそのイメージはパイプステージ１７０，１２６を順番に停動するラッチをクリアする。ラッチをクリアすると直ぐに、新しいｇｏ＿ｔｏｋｅｎを受け取る。

【００６０】図６を詳細に参照すると、ｉｆｕ＿ｆｐ＿ｇｏ＿ｄｅｃはＣＰＵデコーダパイプステージ１６６からのｇｏ＿ｔｏｋｅｎ信号である。そのＣＰＵデコーダパイプステージ１６６が知らせることは、デコーダパイプステージ１６６内の命令が順調に復号され、デコーダパイプステージが停動されていないことである。同じ方

法で、ｆｐｕ＿ｆｐ＿ｇｏ＿ｄｅｃ信号は浮動小数点ユニットデコーダパイプステージ１７８からのトークン信号である。その浮動小数点ユニットデコーダパイプステージ１７８が知らせることは、デコーダパイプステージ１７８内の浮動小数点命令が順調に復号され、デコーダパイプステージ停動状態が存在しないことである。このトークン信号を生成する前に復号を完了するので、これらの信号は、当該クロックサイクル内で比較的遅れて他のパイプラインに到達する。その結果、これらの信号は受信パイプラインパイプステージ内で即座にラッチされる。例えば、ｉｆｕ＿ｆｐ＿ｇｏ＿ｄｅｃはラッチ２４０によってラッチされ、ｆｐｕ＿ｆｐ＿ｇｏ＿ｄｅｃ信号はラッチ２４２によってラッチされる。組合せ論理２４４は、ライン２４６上にｉｆｐ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｅ１信号を生成するために、ラッチ２４４内でラッチされた信号に応答する。そのｉｆｐ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｅ１信号は、パイプステージ１７２へ命令を送るために実行パイプステージ１７０をトリガする。命令がパイプステージ１７２を離れると直ぐに、ｉｆｕ＿ｆｐ＿ｌｅａｖｉｎｇ＿ｅ１信号がライン２４７上で生成される。そのｉｆｕ＿ｆｐ＿ｌｅａｖｉｎｇ＿ｅ１信号は、組合せ論理２４４をリセットしてｉｆｕ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｅ１信号を非作動化する。そのため、パイプステージ１７０にロードされた次の命令が、パイプステージ１７０を退出可能になる以前に、別のｆｐｕ＿ｆｐ＿ｇｏ＿ｄｅｃトークン信号を必要とするはずである。

【００６１】同様の方法では、ｉｆｕ＿ｆｐ＿ｇｏ＿ｆ１信号はラッチ２４０によって組合せ論理２４８に出力される。組合せ論理２４８はライン２５０上にｆｐｕ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｆ１信号を生成する。そのｆｐｕ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｆ１信号は、パイプステージ１２８へ命令を送るために、ＦＰＵのパイプステージ１２６をトリガする。ひとたび命令がパイプステージ１２６を離れると、パイプステージ１２６はライン２５２上にｆｐｕ＿ｆｐ＿ｌｅａｖｅ＿ｆ１信号を生成する。そのｆｐｕ＿ｆｐ＿ｌｅａｖｅ＿ｆ１信号によって組合せ論理２４８がｆｐｕ＿ｆｐ＿ｍａｙ＿ｌｅａｖｅ＿ｆ１信号を非作動化させる。そのため、パイプステージ１２６にロードされた次の命令は、パイプステージ１２６を離れることが可能になる前に、別のｉｆｕ＿ｆｐ＿ｇｏ＿ｄｅｃトークン信号を必要とするだろう。

【００６２】図５に示す同期化機構の結果として、同一命令がデコーダパイプステージ１６８と浮動小数点デコーダパイプステージ１７８に入るので、ＣＰＵパイプライン１６０内のパイプステージ１６８，１７０とＦＰＵパイプライン１６２のパイプステージ１７８，１２６との間で同期化を失う可能性のある唯一の方法は、それぞれのデコーダパイプステージ１６８，１７８内での遅延の結果としてである。図６で示された機構は、命令がパイプステージ１７０，１２６にそれぞれ進行する時までに、ＣＰＵパイプライン１６０をＦＰＵパイプライン１６２と再同期化する。それにより、命令がこれらのパイプステージを離れる準備をした時に、二つのパイプラインが再同期化される。

【００６３】次の方程式は、図示された同期化論理の演算を説明している。

【００６４】ifu_fp_may_leave_e1 = fpu_fp_go_dec_q | ifu_token_received_q
ifu_token_received_d = ifu_fp_may_leave_e1 & ～ifu_fp_leaving_e1
ifu_fp_leaving_e1 = ifu_fp_valid_e1 & ifu_fp_may_leave_e1 & ～lsu_stall_e3

次の方程式は、ｇｏ－ｔｏｋｅｎがどのようにしてＣＰＵパイプライン１６０から生成されるかを説明している。

【００６５】ifu_fp_go_dec = ifu_fp_valid_dec & ～ifu_dec_stall_cond

即ち、デコーダパイプステージ１６８内の有効な浮動小数点命令上にデコーダパイプステージ停動状態が検出されない限り、ｇｏ－ｔｏｋｅｎは常にパイプライン１６２に信号として送られるだろう。

【００６６】方程式の次の組は、ＦＰＵパイプライン１６２からＣＰＵパイプライン１６０までｇｏ－ｔｏｋｅｎｓを生成するために必要な論理を説明している。

【００６７】fpu_fp_may_leave_f1 = ifu_fp_go_dec_q | fpu_token_received_q
fpu_token_received_d = fpu_fp_may_leave_f1 & ～fpu_fp_leaving_f1
fpu_fp_leaving_f1 + fpu_fp_image_valid_f1 & fpu_fp_may_leave_f1 & ～fpu_stall_f4
fpu_fp_go_dec = fpu_fp_image_valid_dec & ～fpu_go_dec_stall_cond

ひとたび一つの命令がＣＰＵパイプステージ１７０及びＦＰＵパイプステージ１２６を終了させると、複数の命令が通常二つのパイプラインのうち残ったパイプステージを介してロックステップ内で実行されるべきである。

【００６８】しかしながら、ＣＰＵパイプライン１６０とＦＰＵパイプライン１６２とに相互に同期化を失わせることのできる他の種類の停動状態がＣＰＵ内に存在する。この付加的なタイプの停動状態はロード／ストアユニット停動状態である。ロード／ストアユニット停動状態は、ＣＰＵパイプライン１６０のパイプステージ１７４で発生し、その発生原因は、例えば、オペランドキャッシュを避けるロード／ストア命令である。図７は、これらの状態の下に、ＣＰＵパイプライン１６０とＦＰＵパイプライン１６２とを停動及び再同期化するために用いられる論理回路を示す。特に、図７に示す論理２８０は、二つのパイプラインを再同期化するために用いられる。

【００６９】ロード／ストアユニット停動状態が発生する時、ｌｓｕ＿ｓｔａｌｌ＿ｅ３信号がライン２８２でアサート（表明）される。この信号がアサートされる時に、パイプステージ１７４と、ＣＰＵパイプライン１６２における以前の全てのパイプステージ１６６，１７０，１７２が即座に停動される。また、ライン２８２上のｌｓｕ＿ｓｔａｌｌ＿ｅ３信号もインタフェース５６を横切って論理２８０に送られる。ｌｓｕ＿ｓｔａｌｌ＿ｅ３信号は、ＣＰＵパイプライン１６０を停動するクロックサイクルの間に、ラッチ２８４内でラッチされる。しかしながら、ｌｓｕ＿ｓｔａｌｌ＿ｅ３がアサートされるクロックサイクルの間に、ＦＰＵパイプライン１６２は実行を継続する。次のクロックサイクルにおいて、ラッチされた停動信号はＦＰＵパイプライン１６２のパイプステージ１３２に送られ、ＦＰＵパイプライン１６２がＦＰＵパイプステージ１７８，１２６，１２８，１３０，１３２を即座に停動する。同一クロックサイクルの間に、ラッチ２８４からのライン２８６上の停動信号を使ってラッチ２８８，２９０，２９２のラッチ機能を不能化し、マルチプレクサ２９４，２９６，２９８を制御してライン３０１，３０３，３０５上でラッチされたデータをそれぞれ選択する。それによって、デコーダパイプステージＦＣＭＰからのｇｏ＿ｔｏｋｅｎ（２つの浮動小数点レジスタを比較するＦＰＵ命令）の状態と、実行パイプステージ１２８からの例外情報とを維持する。ＦＰＵ実行ユニットからのデータのラッチは、ＣＰＵパイプライン１６０内の実行パイプステージと交信し、ＦＰＵパイプライン１６２の停動時にこのデータが失われないことを保証する。これは次のことを保証している。即ち、ライン２９５，２９７，２９９上でＣＰＵパイプライン１６０に送信中のデータがＦＰＵパイプステージからのデータであり、そしてＣＰＵパイプライン実行に関してＦＰＵパイプライン実行が前進したクロックサイクルの間に、そのデータが生成されたことである。図７に示す論理の結果として、ロード／ストアユニット停動状態に起因してＣＰＵが停動する時に、浮動小数点ユニットは、ＣＰＵパイプラインに関して一つのパイプステージだけ進む。しかし、ＦＰＵパイプラインは次のクロックサイクル時に停動され、その代わりに、通常、ＣＰＵパイプラインへ伝送された全てのデータが記憶される。

【００７０】ライン２８２上のｌｓｕ－ｓｔａｌｌ＿ｅ３信号が非作動化される時、ＣＰＵパイプライン１６０は即座に実行を開始し、現在停動されたＦＰＵパイプライン１６２に関して一つのパイプステージだけ進む。このクロックサイクルの間に、ＣＰＵパイプステージは、ＦＰＵの停動時に記憶された状態でそれぞれラッチ２８８，２９０，２９２からのライン２９５，２９７，２９９上のデータを読み取る。ラッチ２８４の結果として、次のクロックサイクル上で、ライン２８５，２８６上の

停動信号が非作動化される。これによって、ＦＰＵパイプライン１６２が即座に再開始させられる。しかしながら、ＣＰＵパイプライン１６０がＦＰＵパイプライン１６２の再開始の１クロックサイクル前に再開始されるので、ライン２８６，２８５上の停動信号が非作動化される時に、二つのパイプラインは、ロード／ストアユニット停動状態が発生した以前に有していたのと同じ関係に再同期化され、データロスは発生しない。ライン２８６上の信号が非作動化される時、マルチプレクサ２９４はライン３００上のｇｏ－ｔｏｋｅｎ信号を選択し、マルチプレクサ２９６はライン３０２上のデータ信号を選択し、マルチプレクサ２９８はライン３０４上の例外信号を選択する。そのため、ＣＰＵパイプライン１６０は再びＦＰＵパイプライン１６２から現行信号を受け取る。このようにして、二つのパイプラインの動作が再同期化され、浮動小数点命令の実行が継続する。

【００７１】命令がＣＰＵパイプライン１６０の書き戻しパイプステージ１７６に入る時に、及び、命令がＦＰＵパイプライン１６２のパイプステージ１３２に入る時に、ＣＰＵパイプライン１６０とＦＰＵパイプライン１６２との間の最終同期化点が発生する。精密な例外モデルを維持するために、例えば、純粋なＣＰＵ命令の場合には、ＣＰＵからＦＰＵまでのキャンセル命令がｉｆｕ＿ｆｐｕ＿ｃａｎｃｅｌ＿ｗｂ信号としてライン３０６上で送られる。命令がパイプステージ１７６でＣＰＵによってキャンセルされなかったならば、浮動小数点パイプライン１６０は実行を継続する。ＦＰＵパイプライン１６２がキャンセル命令を受け取る時、ＦＰＵ５４は、ＦＰＵパイプステージ１７８，１２６，１２８，１３０，１３２内で実行する全ての命令をキャンセルする。

【００７２】本発明の結果として、ＣＰＵコア５１内の唯一のオプションではあるが、ＦＰＵ５４をＣＰＵ５２にインターフェイスできる。そのため、ＣＰＵ及びＦＰＵは密接に結合されて高性能処理能力を維持する。更に、ＣＰＵパイプラインとＦＰＵパイプラインとが所定数のサイクルによって相互に関してスリップするように拘束されているので、ＣＰＵパイプラインとＦＰＵパイプラインとの密接な結合がマイクロコンピュータ５０内の精密な例外モデルを維持する。

【００７３】前述したように、本発明は１つの単一集積回路において実行可能である。

【００７４】こうして、本発明の少なくとも１つの例示的な実施の形態について記述したが、当該技術分野における当業者にとって種々の変更、変形、及び改良は容易に可能である。この種の変更、変形、及び改良は、本発明の精神及び範囲に含まれるものである。従って、今までの記述は単に一例に過ぎず、限定を意図するものではない。本発明は請求の範囲における定義と、それと等価なことによってのみ限定される。

【００７５】

【発明の効果】本発明によれば、ＣＰＵコア内の唯一のオプションではあるが、ＦＰＵをＣＰＵにインターフェイスできる。そのため、ＣＰＵ及びＦＰＵは密接に結合されて高性能処理能力を維持する。更に、ＣＰＵパイプラインとＦＰＵパイプラインとが所定数のサイクルによって相互に関してスリップするように拘束されているので、ＣＰＵパイプラインとＦＰＵパイプラインとの密接な結合がマイクロコンピュータ内の精密な例外モデルを維持する。

【図面の簡単な説明】

【図１】選択的（オプション）な浮動小数点プロセッサ（ＦＰＵ）を含んだ本発明によるマイクロコンピュータを示す図である。

【図２】図１のマイクロコンピュータ内で使用可能なＦＰＵとＣＰＵとの間で浮動小数点ユニットとインタフェースとを示すブロック図である。

【図３】ＣＰＵ実行パイプラインと、ＦＰＵ実行パイプラインと、図１のマイクロコンピュータの各パイプライン内でパイプステージ間の関係とを示す図である。

【図４】図１のマイクロコンピュータ内のＣＰＵとＦＰＵとの間のインタフェースのブロック図であり、二つのパイプラインを同期化するために用いられた電気回路の構成部分と信号とを示す。

【図５】図４のＣＰＵプリデコーダステージ命令バッファ（緩衝）機構における更に詳細な論理的ブロック図である。

【図６】図４のデコーダ／Ｅ１－Ｆ１ステージ同期化論理における更に詳細な論理的なブロック図である。

【図７】図４の一部分の論理的なブロック図であり、ロード／ストアユニットＥステージ停動及び再同期化の論理を示す。

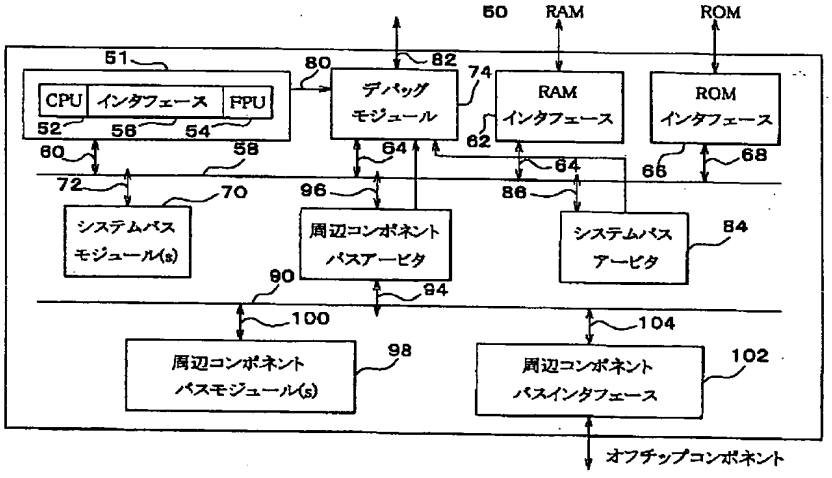
【符号の説明】

５０　単一チップマイクロコンピュータ
５１　中央処理装置コア
５２　整数中央処理装置（ＣＰＵ）
５４　浮動小数点処理装置（ＦＰＵ）
５６　インタフェース
５８　システムバス
６０　データリンク
６２　ＲＡＭインタフェース
６４　データリンク
６６　ＲＯＭインタフェース
６８　データリンク
７０　システムバスモジュール
７２　データリンク
７４　デバッグモジュール
７６　データリンク
８０　データリンク
８２　デバッグリンク
８４　システムバスアービタ
８６　データリンク
８８　データリンク
９０　周辺コンポネントバス
９２　周辺コンポネントバスアービタ
９４　データリンク
９６　データリンク
９８　周辺コンポネントバスモジュール
１００　データリンク
１０２　周辺コンポネントバスインタフェース
１０４　データリンク
１１０　モジュール
１１２　モジュール
１１４　モジュール
１１６　実行パイプステージ
１１８　実行パイプステージ
１２０　実行パイプステージ
１２２　実行パイプステージ
１２４　モジュール
１２６　実行パイプステージ
１２６Ａ　バッファ
１２６Ｂ　ラッチ
１２８　実行パイプステージ
１２８Ａ　バッファ
１３０　実行パイプステージ
１３０Ａ　バッファ
１３２　実行パイプステージ
１３２Ａ　バッファ
１３４　実行パイプステージ
１３４Ａ　バッファ
１３６　モジュール
１４２　完了バス
１４４　ディスパッチバス
１５０　３２ビット命令
１５２　６４ビット命令
１５４　６４ビット命令
１５６　制御信号インタフェース
１６０　実行パイプライン
１６２　実行パイプライン
１６４　命令フェッチパイプステージ
１６６　プリデコーダパイプステージ
１６８　デコーダパイプステージ
１７０　実行パイプステージ
１７０Ａ　バッファ
１７０Ｂ　ラッチ
１７２　実行パイプステージ
１７２Ａ　バッファ
１７２Ｂ　ラッチ
１７４　実行パイプステージ
１７４Ａ　バッファ
１７４Ｂ　ラッチ

１７４Ｃ	ラッチ	＊２４７	ライン
１７６	書き戻しパイプステージ	２４８	組合せ論理
１７６Ａ	バッファ	２５０	ライン
１７８	浮動小数点デコーダパイプステージ	２５２	ライン
１８０	浮動小数点書き戻しステージ	２８０	論理
１８０Ａ	バッファ	２８２	ライン
２００	プリデコーダ論理回路	２８４	ラッチ
２０２	ラッチ	２８５	ライン
２０４	ラッチ	２８６	ライン
２０６	マルチプレクサ	２８８	ラッチ
２０８	プリデコーダバッファ	２９０	ラッチ
２１０	接続部	２９２	ラッチ
２１２	プリデコーダ	２９４	マルチプレクサ
２１４	マルチプレクサ	２９５	ライン
２１６	接続部	２９６	マルチプレクサ
２２０	出力	２９７	ライン
２２２	出力	２９８	マルチプレクサ
２２４	命令フェッチユニットデコーダバッファ	２９９	ライン
２２６	デコーダバッファ	３００	ライン
２２７	出力	３０１	ライン
２２８	出力	３０２	ライン
２４０	ラッチ	３０３	ライン
２４２	ラッチ	３０４	ライン
２４４	組合せ論理	３０５	ライン
２４６	ライン	＊３０６	ライン

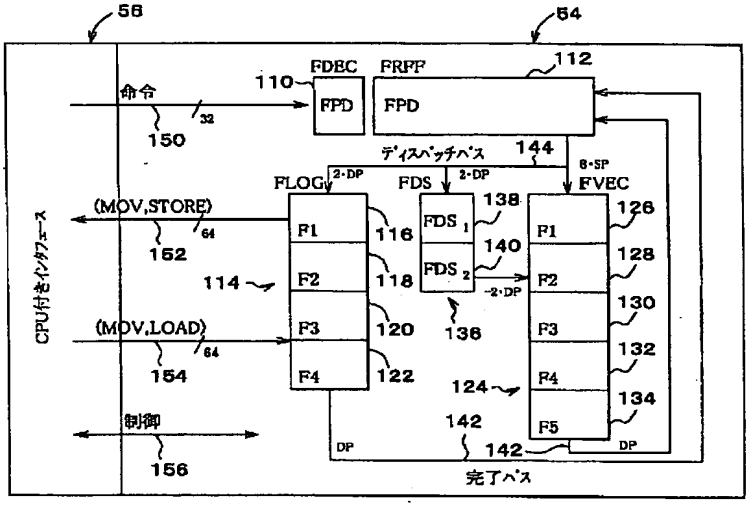
【図１】

図　１



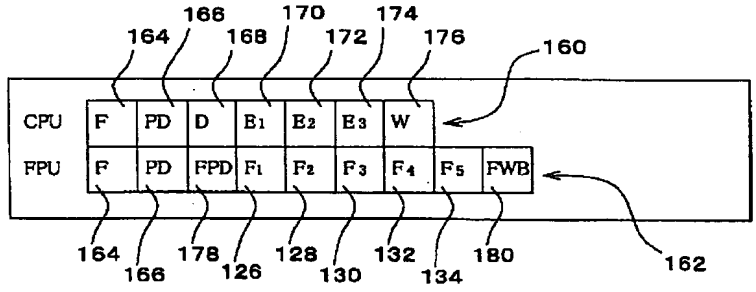
【図２】

図 2



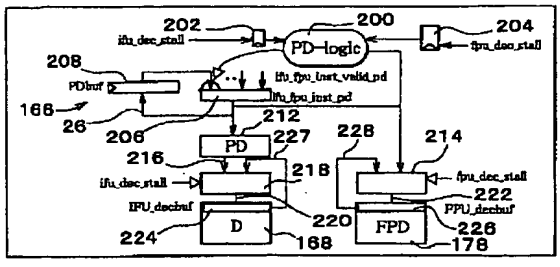
【図３】

図 3



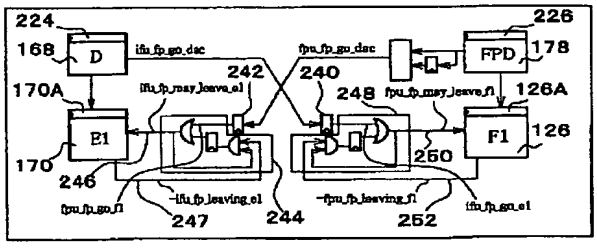
【図５】

図 5



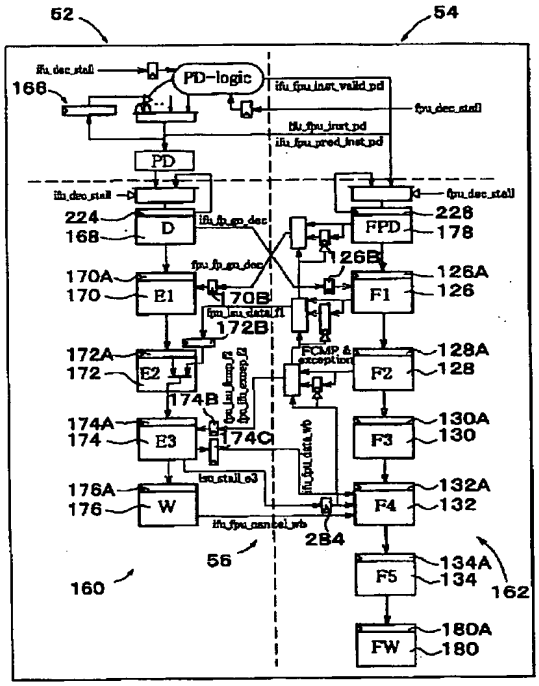
【図６】

図 6



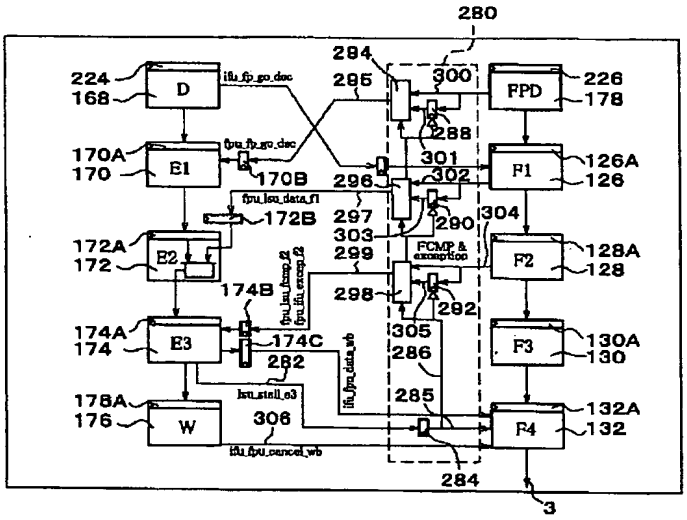
【図４】

図 4


62
54
PD-logic
166
PD
224
168
D
170A
170
E1
170B
172B
172A
172
E2
174B
174A
174
E3
174C
176A
176
W
56
160
226
178
FPD
126B
126A
126
F1
128A
128
F2
130A
130
F3
132A
132
F4
284
134A
134
F5
162
180A
180
FW

【図７】

図 7


280
294
224
168
D
295
300
FPD
226
178
288
170A
170
E1
301
302
F1
126A
126
170B
296
172B
297
303
280
304
172A
172
E2
299
F2
128A
128
174B
298
292
174A
174
E3
174C
305
286
F3
130A
130
178A
176
W
306
285
F4
132A
132
284
3